Observabilidade no ambiente de Computação em Névoa

Breno Costa, Aletéia P. F. Araújo

[1]Departamento de Ciencia da Computação – Universidade de Brasília (UnB)
Brasília – DF – BraSil

brenogscosta@gmail.com, aleteia@unb.br

Resumo. *A observabilidade em sistemas distribuídos é a capacidade de entender o estado interno de um sistema a partir de seus dados externos. Uma maior observabilidade permite rapidez na análise de problemas encontrados em tempo de execução, auxiliando a manutenção dos níveis de serviços acordados. Computação em Névoa fornece recursos computacionais próximos aos usuários e tem como características a restrição de recursos e a heterogeneidade dos dispositivos e conexões. Este trabalho se propõe a definir e caracterizar os desafios de aumentar a observabilidade de sistemas em um ambiente de Névoa. Para isso, um ambiente de testes real foi criado e configurado apenas com soluções de código aberto. Por meio de um caso de uso de Internet das Coisas, pôde-se medir a sobrecarga característica das soluções atuais de observabilidade e discutir acerca dos desafios desta implantação na Névoa.*

Abstract. *Observability in distributed systems is the ability to understand the internal state of a system from its external data. Greater observability allows faster analysis of issues encountered at runtime, helping to meet agreed service levels. Fog Computing provides computational resources close to users and it is characterised by resource constraints and heterogeneity of devices and connections. This work proposes to define and characterise the challenges of increasing the observability of systems in a fog environment. For this, a real test environment was created and configured only with open source solutions. Through an Internet of Things use case, it was possible to measure the characteristic overhead of current observability solutions and discuss the challenges of such implementation in a Fog environment.*

1. Introdução

Computação em Névoa é um paradigma computacional que complementa a Nuvem, fornecendo recursos computacionais na borda da rede, mais próximos aos usuários. Por ser distribuída, a Névoa tem de lidar com a heterogeneidade dos links de rede e capacidade baixa de processamento de seus nós [Bonomi et al. 2012].

Para aumentar a observabilidade de um sistema distribuído, é preciso instrumentá-lo de várias formas. São três os domínios de instrumentação da Observabilidade: *logs*, métricas e *traces*. Logs são linhas de texto que um aplicativo gera em pontos discretos durante a execução do código. Métricas são valores representados como contagens ou medidas que calculamos ou agregamos ao longo de um período de tempo. Um *trace* é uma representação de uma requisição, à medida que ela flui por um sistema distribuído. Ele registra o tempo gasto em cada passo do começo ao fim de sua execução [Costa et al. 2022].

Essas informações quando analisadas tempestivamente permitem depurar com mais eficácia o sistema e a infraestrutura, agilizando a solução de problemas e o retorno do sistema a um estado adequado. Para tanto, é necessário manter um fluxo de coleta dos dados de telemetria para analisá-los

em conjunto e tomar decisões apropriadas quando necessário. Em um ambiente de Névoa, há vários desafios a serem enfrentados devido à incerteza da conectividade e da restrição de recursos.

Este trabalho propõe uma caracterização da observabilidade em ambiente de Computação em Névoa. As principais contribuições são:

- Uma descrição detalhada dos domínios da observabilidade, caracterizando-os para o uso no ambiente de Computação em Névoa;
- Uso de um caso de uso de Internet das Coisas (IoT, do termo em inglês) em uma ambiente de testes real;
- Avaliação da sobrecarga que o aumento da observabilidade pode trazer ao ambiente de Névoa em comparação com os benefícios que podem ser alcançados.

2. Observabilidade

A observabilidade é uma característica dos sistemas de fornecer informações sobre seus estados internos por meio de saídas externas. Quanto maior a observabilidade, mais fácil entender os comportamentos atuais e passados do sistema [Karumuri et al. 2021]. Este conhecimento sobre o sistema, desde o momento em que está disponível, permite uma atuação adequada sobre aquele sistema quando necessário.

Os sistemas IoT que executam na Névoa são caracterizados por terem uma organização mais distribuída, heterogeneidade de dispositivos físicos e redes, e incerteza de conectividade, causada pela mobilidade dos dispositivos, instabilidades da rede e esgotamento da bateria [Iorga et al. 2018]. Esse cenário é bastante diverso de sistemas em nuvem, suportados por servidores homogêneos ricos em recursos, fonte de alimentação contínua e conexões de rede redundantes e estáveis.

2.1. Domínios de Instrumentação da Observabilidade: Métricas, Logs e Traces

A observabilidade deve ser instrumentada. Existem três domínios de instrumentação de Observabilidade: métricas, logs e *traces* [Karumuri et al. 2021]. Cada domínio de instrumentação contribui para a observabilidade de um sistema de maneira diferente.

Métricas estão mais relacionadas ao desempenho de um sistema. São valores numéricos coletados em um ponto do tempo e sua coleta pode ser caracterizada como uma série temporal[Karumuri et al. 2021]. Existem muitas métricas que podem ser coletadas sobre a infraestrutura e o aplicativo. Por exemplo: percentual de uso da CPU, vazão da rede 5G em Mbps, etc. Diferentes métricas podem levar a diferentes atuações. Uma vazão de rede que caiu abaixo de um determinado limite pode sinalizar que as transferências de dados devem ser adiadas para outro momento.

Os **logs** são arquivos de texto não estruturados ou semiestruturados, relatando eventos relevantes e informações contextuais, cuja instrumentação é feita geralmente em tempo de desenvolvimento[Karumuri et al. 2021]. Por exemplo, o desenvolvedor decidiu gravar no log as transações bem-sucedidas e os detalhes do erro. Usando um sistema de IoT, podemos ter nos logs informações relacionadas à qualidade de serviço da conexão de rede a cada segundo, coordenadas geográficas do dispositivo etc. Analisando esses dados por um período de tempo, podemos descobrir locais onde a vazão da rede é baixa em muitos momentos do dia. Essa descoberta pode levar a uma investigação sobre a cobertura da rede 5G que pode ser discutida com a operadora de rede.

Traces são registros de chamadas feitas pelo sistema. Permitem observar o tempo gasto em cada chamada de serviço e a sequência de chamadas do início ao fim de uma solicitação[Karumuri et al. 2021]. A análise de *traces* pode mostrar quais chamadas de serviço estão demorando mais na composição do tempo de resposta de um sistema. Eles também podem mostrar solicitações que não foram concluídas corretamente. A atuação no primeiro caso pode ser uma

otimização de código entregue como uma nova versão do sistema. Neste último caso, um melhor gerenciamento de erros pode tornar o aplicativo mais resiliente ao processamento de solicitações malformadas.

Métricas, logs e traces contribuem de forma independente para aumentar a observabilidade de um sistema. Como exemplificado, cada um deles fornece um tipo diferente de informação, possibilitando uma atuação complementar. As métricas fornecem informações objetivas sobre a interface externa de um sistema, por exemplo, taxa de transferência de upload de vídeo. Elas permitem uma tomada de decisão rápida em resposta àquelas medições que estão fora de um intervalo regular declarado. O volume de dados gerado usualmente é baixo e estável. Os *logs* fornecem informações internas sobre eventos de falha, como mensagens de erro específicas, mensagens de tratamento de exceções, erros de tempo de execução. Eles podem fornecer as informações necessárias para acelerar a análise da causa raiz, ajudando a equipe de manutenção a melhorar o tratamento de erros e retornar o sistema a um estado saudável. Os *traces* fornecem detalhes sobre o fluxo interno de informações, incluindo a sequência e o desempenho de cada chamada de serviço necessária para processar uma solicitação. Esses dados podem ser visualizados como um grafo e um caminho crítico pode ser criado a partir dele, permitindo escrutinar a dependência entre os componentes de um sistema.

É possível conectar os três domínios no momento em que cada informação foi gerada. Quando é viável relacionar dois ou três deles em uma mesma análise, surgem mais oportunidades de atuação. Os dados de *log*, quando comparados às métricas coletadas durante o mesmo período de tempo, permitem uma inspeção mais abrangente dos problemas, reunindo as visões externa e interna do sistema simultaneamente. Um *trace* pode ser visto como um detalhamento de uma métrica de tempo de resposta, permitindo identificar os componentes onde uma melhoria no atraso de processamento ou comunicação pode resultar em um tempo de resposta final menor.

Ter mais domínios de instrumentação disponíveis significa um nível mais alto de observabilidade. Além disso, além do valor independente de cada domínio, existe um valor adicional na análise cruzada entre os domínios. Então, ao invés de ter uma fórmula onde a observabilidade é uma função da soma de seus domínios de instrumentação como na Equação (I),

(I) Observabilidade = Métricas + Logs + Traces

É preciso adicionar as interações entre eles também, como na Equação (II):

(II) Observabilidade = Métricas + Logs + Traços + (Métricas X Logs X Traces)

2.2. Ciclo de vida dos dados de observabilidade

Para coletar informações de cada domínio de instrumentação e aumentar a observabilidade de um sistema, é necessário estar ciente do seguinte ciclo de vida dos dados, representado na Figura 1: 1. geração de dados; 2. armazenamento local; 3. transmissão de dados; 4. agregação de dados; 5. consulta aos dados; 6. Armazenamento definitivo.

1. Geração de Dados - Na fase inicial do ciclo de vida dos dados de observabilidade, os dados são criados. Isso pode acontecer de várias maneiras, de acordo com o domínio de instrumentação em vigor. As métricas podem ser adquiridas do sistema operacional por meio de chamadas de sistema que relatam a quantidade de recursos disponíveis (por exemplo, CPU, memória, armazenamento em disco). Os logs são gravados de acordo com o fluxo de evento específico que foi instrumentado para ser registrado em texto. Eventos bem-sucedidos, como solicitações http atendidas (código 200) ou pilha de chamada de função, em caso de exceções de tempo de execução detectadas no código. Quando previamente instrumentados, os *traces* podem ser sinalizados por chamadas de API específicas que registram a sequência de chamadas e o tempo de cada chamada.

Figure 1. Ciclo de vida dos dados de observabilidade.

2. Armazenamento local de dados aguardando coleta - Nesta fase, os dados de observabilidade foram gerados pelo sistema e agora estão em armazenamento local aguardando coleta ou remoção. Como os dados novos são sempre adicionados ao conjunto anterior, a tendência é que fiquem cada vez maiores com o passar do tempo. Portanto, para não ficar sem recursos de armazenamento, uma política de remoção de dados deve estar em vigor. Embora as métricas possam ser constantes em termos de volume de dados, logs e *traces* têm uma variabilidade maior. Essa característica traz o desafio de monitorar o uso do armazenamento e atuar quando há risco de falta de recursos.

3. Transmissão de dados até o ponto de agregação e análise - Para permitir a agregação e análise de dados, os dados gerados pelo sistema devem ser coletados usando as conexões de rede existentes. Mas essas conexões também são utilizadas pelo sistema para receber e responder às solicitações do usuário. Portanto, gerenciadores de dados de observabilidade podem de alguma forma competir por recursos de rede com o próprio sistema e interferir negativamente nos SLAs[Popiolek and Mendizabal 2012]. Principalmente em relação a logs e traces que comumente geram maior volume de dados.

4. Agregação de dados de acordo com o tipo de dados e uso - As métricas podem ser compreendidas como uma série temporal e um banco de dados de séries temporais deve ser usado para armazená-las. Mas logs e *traces* são estruturados de forma diferente e se beneficiarão de outras soluções de armazenamento. Karumuri et al. [Karumuri et al. 2021] analisaram dados de observabilidade em um ambiente de nuvem e propuseram que os logs fossem armazenados em um banco de dados colunar e os traces em um banco de dados de grafos. Essas ferramentas fornecem melhor acesso e consultas mais rápidas a esses tipos de dados. Assim, o serviço de agregação de dados de observabilidade deve considerar cada domínio de instrumentação de forma independente para armazená-los, mas permitindo que análises cruzadas sejam feitas entre eles.

5. Consulta aos dados para tomada de decisão - Como os dados são agregados e disponibilizados para o servidor de observabilidade, é possível consultá-los e tomar decisões e ações de acordo. Os dados de observabilidade tendem a dar respostas mais relevantes em consultas realizadas sobre os dados mais recentes (últimas 24 horas). Assim, é importante garantir o acesso rápido a essa janela de tempo e fornecer mecanismos para enviar os dados fora desse intervalo para um armazenamento de longo prazo.

6. Armazenamento de dados de longo prazo - Depois que os dados atingem a janela de tempo em que há baixa demanda de consultas, eles podem ser movidos para um armazenamento de longo prazo, onde são esperados maiores volumes de dados e onde são processadas consultas que consideram o histórico de dados.

3. Considerações Finais

Este trabalho definiu as características e desafios do aumento da observabilidade dos sistemas em um ambiente de Computação em Névoa. Quanto maior a Observabilidade, maior a probabilidade de entender a causa-raiz de problemas no ambiente de execução e de solucionar esses problemas mais rapidamente, garantindo a manutenção dos SLAs.

A observabilidade pode ser instrumentada por meio de métricas, logs e traces. Cada um destes conjuntos de informação de observabilidade possui características específicas acerca dos tipo de dado, volume e frequência de geração. São necessários mecanismos específicos para coletar e armazenar essas informações de forma simultânea e centralizada, implementando um ciclo de vida que permita a tomada rápida de decisões, inclusive de forma autônoma, e que atenda às restrições impostas pelo ambiente de Névoa.

Por fim, um caso de uso de IoT será implementado em um ambiente de testes real de Computação em Névoa, configurado com ferramentas de observabilidade de código aberto, de forma a caracterizar a sobrecarga que essas ferramentas adicionam à infraestrutura e entender os desafios específicos.

References

Bonomi, F., Milito, R., Zhu, J., and Addepalli, S. (2012). Fog computing and its role in the internet of things. In *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, MCC '12, pages 13–16, New York, NY, USA. ACM.

Costa, B., Bachiega Jr, J., Carvalho, L. R., Rosa, M., and Araujo, A. (2022). Monitoring fog computing: A review, taxonomy and open challenges. *Computer Networks*, page 109189.

Iorga, M., Feldman, L., Barton, R., Martin, M. J., Goren, N. S., and Mahmoudi, C. (2018). Fog computing conceptual model.

Karumuri, S., Solleza, F., Zdonik, S., and Tatbul, N. (2021). Towards observability data management at scale. *ACM SIGMOD Record*, 49(4):18–23.

Popiolek, P. F. and Mendizabal, O. M. (2012). Monitoring and analysis of performance impact in virtualized environments. *Journal of Applied Computing Research*, 2:75–82.